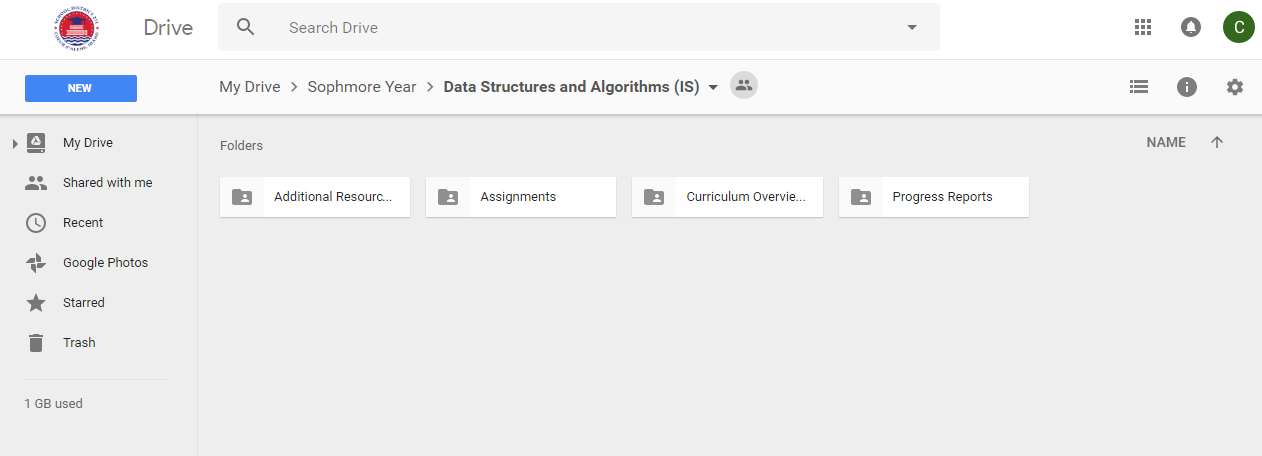
This document is a broader description of my IS progress. All assignments can be found in the “Assignment” folder and descriptions of learned *content* by term can be found in the “Term 1”, “Term 2”, and “Term 3” folders along with this document. Below is an image to nativage different files within the Data Structures and Algorithms (IS) folder. Assignment documents can be found in the Assignments folder. *Completed* assignments can be found on [GitHub](https://github.com/Crista2019/Independent-Study-2016-2017) in the form of a python file or pdf.





**Summary:**

The computer science/technology independent study done during my sophomore year at Lake City High School, concerning "Data Structures and Algorithms." Specifically, this is a semester-long student-led study over the formats and methods computers use to store/organize data. This allows the computer to access different types of information, stored in ways corresponding to that data's purpose. This IS utilizes the programming language python to demonstrate concepts and complete assignments. The curriculum is a combination of multiple resorces, such as "Problem Solving with Algorithms and Data Structures using Python (https://interactivepython.org/runestone/static/pythonds/index.html), online tutorials (https://hackr.io/tutorials/learn-data-structures-algorithms and http://www.algolist.net), and additional information from a variety of different sources, including Casey Falk (software developer at Amazon) and Bill Keylon (LCHS computer science teacher). The 18 week semester is divided into three 6 week long terms/units: *Exploration into Data Structures and Algorithms, Computational Complexity (‘Big O’ Analysis)*, and a final *Synthesis Project* which is a demonstration of good data structure and algorithm use designed by the student.

<https://github.com/Crista2019/Independent-Study-2016-2017>

**Term 1**

|  |  |
| --- | --- |
| **Week 1:** | Curriculum/Google Drive folders/Github setup; installation of Sublime Text 3;  Learned about Linked Lists/Arrays  *Linked List Assignment #1*  *Array List Assignment #1* |
| **Week 2:** | Learned about Stacks  *Stacks Assignment #2* |
| **Week 3:** | Learned about recursion  *Recursion lesson & assignment #3* |
| **Week 4:** | Implemented recursion into each example function of previous assignments (*Linked List Assignment #1.1 Stacks Assignment #2.1)*  Learned about queues  *Queues Assignment #4* |
| **Week 5:** | Learned about trees  *Trees Assignment #5* |
| **Week 6:** | Learned about graphs  *Graphs Assignment #6*  Learned about Dictionaries (“Hash Map”)  *Hash Map Assignment #6b*  *Dictionary Practice #6c* |

**Topic(s) covered:** *Exploration into Data Structures and Algorithms*

**Resources Used:**

**W1:** [*https://www.codefellows.org/blog/implementing-a-singly-linked-list-in-python/*](https://www.codefellows.org/blog/implementing-a-singly-linked-list-in-python/)

**W2:** [*https://interactivepython.org/runestone/static/pythonds/BasicDS/WhatisaStack.html*](https://interactivepython.org/runestone/static/pythonds/BasicDS/WhatisaStack.html)[*https://www.tutorialspoint.com/data\_structures\_algorithms/stack\_algorithm.htm*](https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm)

[*https://interactivepython.org/runestone/static/pythonds/BasicDS/ImplementingaStackinPython.html*](https://interactivepython.org/runestone/static/pythonds/BasicDS/ImplementingaStackinPython.html)

**W3:** <https://www.cs.utah.edu/~germain/PPS/Topics/recursion.html>

<http://stackoverflow.com/questions/717725/understanding-recursion>

<http://www.programmerinterview.com/index.php/recursion/explanation-of-recursion/>

<http://softwareengineering.stackexchange.com/questions/25052/in-plain-english-what-is-recursion>

**W4:** <https://repl.it/> (seriously cool browser-based/cloud text editor)

<https://interactivepython.org/runestone/static/pythonds/BasicDS/WhatIsaQueue.html> (3.10-3.14)

**W5:** [www.tutorialspoint.com/data\_structures\_algorithms/tree\_data\_structure.htm](http://www.tutorialspoint.com/data_structures_algorithms/tree_data_structure.htm) <http://www.openbookproject.net/thinkcs/python/english2e/ch21.html> <https://www.cs.cmu.edu/~adamchik/15-121/lectures/Trees/trees.html> <https://en.wikipedia.org/wiki/Trie>

**W6:** <http://interactivepython.org/runestone/static/pythonds/Graphs/toctree.html> <http://www.wou.edu/~jcm/WebPageSpring2014/Postings/GraphADTIntro.pdf> <https://docs.python.org/3/tutorial/datastructures.html#dictionaries>

<http://www.laurentluce.com/posts/python-dictionary-implementation/>

**Term 2**

|  |  |
| --- | --- |
| **Week 1:** | Prepared/refined curriculum for Term 2  Worked on uploading, organizing materials from Term 1 on Google Drive/GitHub/laptop for future reference  Learned about sets and frozen set data type  Reviewed overarching CS concepts  Studied graph traversals; BFS, DFS, A\*, GBF Dijkstra's Algorithm |
| **Week 2:** | Started studying sorting algorithms *(Assignment1T2Sorting.py)* |
| **Week 3:** | **\*Spring Break\*** |
| **Week 4:** | Reviewed tree/graph search algorithms  Finished reviewing sorting algorithms  Researched binary (min/max) heaps and the algorithms associated with that data type  *(Assignment2T2Heaps.py)* |
| **Week 5:** | Studied the use of heuristics in algorithms  Trade-Offs, Use, Pitfalls, Examples  Implemented Dijkstra’s algorithm *(Assignment3T2DijkstrasAlgorithm.py)* |
| **Week 6:** | Continued work on Dijktra’s  Learned about minimum spanning trees  Kruskal Algorithm and Prim’s Spanning Tree Algorithm |

**Topic(s) covered:** *Computational Complexity (“Big O” Analysis)/Continued Data Structure Implementation/ Introduction to Algorithms*

**Resources Used:**

**W1:** <http://www.python-course.eu/sets_frozensets.php> <http://interactivepython.org/runestone/static/pythonds/Introduction/toctree.html> <http://www.geeksforgeeks.org/greedy-algorithms-set-6-dijkstras-shortest-path-algorithm/> <http://cs.stanford.edu/people/abisee/gs.pdf>

**W2:** <http://www.studytonight.com/data-structures/introduction-to-sorting> <https://en.wikipedia.org/wiki/Sorting_algorithm#Popular_sorting_algorithms> <http://www.cs.cmu.edu/~clo/www/CMU/DataStructures/Lessons/lesson8_1.htm>

<https://www.toptal.com/developers/sorting-algorithms>

**W3: SPRING BREAK**

**W4:** <http://interactivepython.org/runestone/static/pythonds/Trees/PriorityQueueswithBinaryHeaps.html?highlight=heaps>

**W5:** <https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm#Algorithm> <https://en.wikipedia.org/wiki/Heuristic_(computer_science)> <https://www.cs.auckland.ac.nz/software/AlgAnim/dijkstra.html> <http://www.geeksforgeeks.org/greedy-algorithms-set-6-dijkstras-shortest-path-algorithm/>

**W6:** <https://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.sparse.csgraph.minimum_spanning_tree.html> <http://www.geeksforgeeks.org/greedy-algorithms-set-2-kruskals-minimum-spanning-tree-mst/> <http://interactivepython.org/courselib/static/pythonds/Graphs/PrimsSpanningTreeAlgorithm.html>

**Term 3**

|  |  |
| --- | --- |
| **Week 1:** | *Review of* Abstract Data Types  Brainstorming *for Synthesis Project* |
| **Week 2:** | *Review of* specific subcategories of data types {e.g. frozen sets, min spanning trees, digraphs et al}  Began organizing thoughts about and programming (completed most of character’s attributes and basic information) for *Synthesis Project*  *(SynthesisProject.py)* |
| **Week 3:** | *Review of* ComSci Core Principles  Created more complex data types such as graphs for maps and binary trees for quests *for Synthesis Project*  *(SynthesisProject.py)* |
| **Week 4:** | *Review of* Algorithm Types  Methods for dynamically creating classes by storing data from user input (assuming user doesn’t know how to define object instance of a class or defining a hashmap of connecting nodes in my dictionary implementation of a graph) *for Synthesis Project* |
| ***Week 5:*** | *Review of Algorithm Analysis {comp.complexity, analysis of loops/recursion, etc}*  User Interface (abstraction) and help() reference menu *for Synthesis Project* |
| **Week 6:** | Content/Vocabulary/Problem Solving quizzes online  Finish up and present project, ensuring functionality by testing program on peers *for Synthesis Project* |

**Topic(s) covered:** *Synthesis Project*

**Resources Used:**

**Week 1:** Notes; <http://interactivepython.org/courselib/static/pythonds/Introduction/WhyStudyDataStructuresandAbstractDataTypes.html> <https://en.wikipedia.org/wiki/List_of_data_structures> <http://scanftree.com/Data_Structure/>

**Week 2:** <http://www.gdeepak.com/IADSA/applicationsofdatastrucuturesalgorithmsinfosys.pdf>

**Week 3:** <http://www.python-course.eu/python3_inheritance.php> <http://carlcheo.com/compsci>

**Week 4:** <http://www.geeksforgeeks.org/fundamentals-of-algorithms/> <https://softwareengineering.stackexchange.com/questions/155639/which-algorithms-data-structures-should-i-recognize-and-know-by-name> <https://www.quora.com/What-are-the-10-algorithms-one-must-know-in-order-to-solve-most-algorithm-problems> <https://pythonschool.net/category/data-structures-algorithms.html>

**Week 5:** <http://www.geeksforgeeks.org/fundamentals-of-algorithms/>

**Week 6:** Google search of “Python data structures/algorithm quizzes”